

Lab 4

1) Create a pod red with redis image and use an initContainer that uses the busybox image and sleeps for 20 seconds

```
apiVersion: v1
kind: Pod
metadata:
  name: redis-pod
spec:
  containers:
  - name: redis-container
    image: redis
    command: ['sh', '-c', 'echo The app is running! && sleep 20']
  initContainers:
  - name: init-myservice
    image: busybox:1.28
    command: ['sleep', '20']
```

2) Create a pod named print-envvars-greeting.

1. Configure spec as, the container name should be print-env-container and use bash image.

2. Create three environment variables:

a. GREETING and its value should be "Welcome to"

b. COMPANY and its value should be "DevOps"

c. GROUP and its value should be "Industries"

3. Use command to echo ["\$(GREETING) \$(COMPANY) \$(GROUP)"] message.

4. You can check the output using <kubctl logs -f [pod-name]>command

```
apiVersion: v1
kind: Pod
metadata:
  name: print-envvars-greeting
spec:
  containers:
  - name: print-env-container
    image: bash
    env:
    - name: GREETING
      value: "Welcome to"
    - name: COMPANY
      value: "DevOps"
    - name: GROUP
      value: "Industries"
    command: ["echo"]
    args: ["$(GREETING)", "$(COMPANY)", "$(GROUP)"]
```

```
controlplane $ kubectl logs -f print-envvars-greeting
Welcome to DevOps Industries
controlplane $
```

3) Create a Persistent Volume with the given specification.

Volume Name: pv-log

Storage: 100Mi

Access Modes: ReadWriteMany

Host Path: /pv/log

```
Editor  Tab 1  +
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-log
spec:
  capacity:
    storage: 100Mi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteMany
  hostPath:
    path: /pv/log
~
```

4) Create a Persistent Volume Claim with the given specification.

Volume Name: claim-log-1

Storage Request: 50Mi

Access Modes: ReadWriteMany

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: claim-log-1
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 50Mi
```

```
controlplane $ kubectl get pvc
NAME          STATUS  VOLUME  CAPACITY  ACCESS MODES  STORAGECLASS  AGE
claim-log-1   Bound   pv-log   100Mi     RWX            standard      68s
controlplane $
```

5) Create a webapp pod to use the persistent volume claim as its storage.

Name: webapp

Image Name: nginx

Volume: PersistentVolumeClaim=claim-log-1

Volume Mount: /var/log/nginx

```
apiVersion: v1
kind: Pod
metadata:
  name: webapp
spec:
  volumes:
    - name: pv-log
      persistentVolumeClaim:
        claimName: claim-log-1
  containers:
    - name: webapp-container
      image: nginx
      volumeMounts:
        - mountPath: "/var/log/nginx"
          name: pv-log
```

6) How many DaemonSets are created in the cluster in all namespaces?

```
controlplane $ kubectl get daemonsets --all-namespaces
NAMESPACE     NAME          DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE SELECTOR  AGE
kube-system    canal         2        2        2      2            2           kubernetes.io/os=linux  35d
kube-system    kube-proxy    2        2        2      2            2           kubernetes.io/os=linux  35d
controlplane $
```

7) What DaemonSets exist on the kube-system namespace?

```
controlplane $ kubectl get daemonsets -n kube-system
NAME          DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE SELECTOR  AGE
canal         2        2        2      2            2           kubernetes.io/os=linux  35d
kube-proxy    2        2        2      2            2           kubernetes.io/os=linux  35d
controlplane $
```

8) What is the image used by the POD deployed by the kube-proxy
DaemonSet

```

controlplane $ kubectl describe daemonset kube-proxy -n kube-system
Name: kube-proxy
Selector: k8s-app=kube-proxy
Node-Selector: kubernetes.io/os=linux
Labels: k8s-app=kube-proxy
Annotations: deprecated.daemonset.template.generation: 1
Desired Number of Nodes Scheduled: 2
Current Number of Nodes Scheduled: 2
Number of Nodes Scheduled with Up-to-date Pods: 2
Number of Nodes Scheduled with Available Pods: 2
Number of Nodes Misscheduled: 0
Pods Status: 2 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels: k8s-app=kube-proxy
  Service Account: kube-proxy
  Containers:
    kube-proxy:
      Image: registry.k8s.io/kube-proxy:v1.26.0
      Port: <none>
      Host Port: <none>
      Command:
        /usr/local/bin/kube-proxy

```

9) Deploy a DaemonSet for FluentD Logging. Use the given specifications:

Name: elasticsearch

Namespace: kube-system

Image: k8s.gcr.io/fluentd-elasticsearch:1.20

```

apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: elasticsearch
  namespace: kube-system
spec:
  selector:
    matchLabels:
      name: fluentd-elasticsearch
  template:
    metadata:
      labels:
        name: fluentd-elasticsearch
    spec:
      containers:
        - name: fluentd-elasticsearch
          image: k8s.gcr.io/fluentd-elasticsearch:1.20

```

10) Create a multi-container pod with 2 containers.

Name: yellow

Container 1 Name: lemon

Container 1 Image: busybox

Container 2 Name: gold

Container 2 Image: redis

```
apiVersion: v1
kind: Pod
metadata:
  name: yellow
spec:
  containers:
  - name: lemon
    image: busybox
    tty: true
  - name: gold
    image: redis
```

11) create a POD called db-pod with the image mysql:5.7 then check the POD status

```
apiVersion: v1
kind: Pod
metadata:
  name: db-pod
spec:
  containers:
  - name: mysql
    image: mysql:5.7
```

```
controlplane $ vim pod.yml
controlplane $ kubectl apply -f pod.yml
pod/db-pod created
controlplane $ kubectl get pod db-pod
NAME      READY   STATUS    RESTARTS   AGE
db-pod    0/1     Error     0           13s
controlplane $
```

12) why the db-pod status not ready?

```
controlplane $ kubectl logs db-pod
2023-01-27 11:42:03+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 5.7.41-1.el7 started.
2023-01-27 11:42:03+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
2023-01-27 11:42:03+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 5.7.41-1.el7 started.
2023-01-27 11:42:03+00:00 [ERROR] [Entrypoint]: Database is uninitialized and password option is not specified
You need to specify one of the following as an environment variable:
- MYSQL_ROOT_PASSWORD
- MYSQL_ALLOW_EMPTY_PASSWORD
- MYSQL_RANDOM_ROOT_PASSWORD
controlplane $
```

13) Create a new secret named db-secret with the data given below.

Secret Name: db-secret

Secret 1: MYSQL_DATABASE=sql01

Secret 2: MYSQL_USER=user1

Secret3: MYSQL_PASSWORD=password

Secret 4: MYSQL_ROOT_PASSWORD=password123

```
apiVersion: v1
kind: Secret
metadata:
  name: db-secret
data:
  MYSQL_DATABASE: c3FsMDEK
  MYSQL_USER: dXNlcjEK
  MYSQL_PASSWORD: cGFzc3dvcmQK
  MYSQL_ROOT_PASSWORD: cGFzc3dvcmQxMjMK
```

14) Configure db-pod to load environment variables from the newly created secret.

Delete and recreate the pod if required.

```
apiVersion: v1
kind: Pod
metadata:
  name: db-pod
spec:
  containers:
    - name: db-container
      image: mysql:5.7
      envFrom:
        - secretRef:
            name: db-secret
```

```
controlplane $ k get po
NAME      READY   STATUS    RESTARTS   AGE
db-pod    1/1     Running   1 (5s ago)  10s
```